CHAPTER

7

ADMINISTERING ACTIVE DIRECTORY

After completing this chapter, you will be able to:

- Identify common Active Directory object types
- Find objects within Active Directory
- Control user access to objects through permissions
- Understand how inheritance works with permissions
- Understand how to publish shared folders
- Move objects within a domain
- Move objects between domains using MoveTree and Netdom
- Understand how to delegate control to users and groups
- Explain the implications of delegating control

Active Directory is the new directory service that ships with Windows 2000. Many people would agree that it is the single-most significant new feature to arrive with this operating system. It affects not only the design aspects of a Windows 2000 network, but also the way administrators will interact with the network.

In the chapter, we will look at some of the ways in which Active Directory will affect administrators directly. We will show how you can use the directory service to query for information in ways that were impossible with previous versions of Windows. We will examine how the architecture of Active Directory will affect the way you control access to Active Directory objects, and how objects can be moved from one location in Active Directory to another.

One of the features that will require planning in your organization is the assignment of administrative permissions to other users or groups. This feature is called **delegation of control**, and it represents a shift in how our networks will be administered. In the past, system administrators bore most of the responsibility of adding users or creating groups, because it was difficult to assign administrative privileges on a granular basis. This situation has changed with Windows 2000 and Active Directory—it is now possible to compartmentalize users or groups and then to assign permissions over that specific group of computers or people.

This chapter focuses on some of the common tasks administrators will be asked to perform when working with Active Directory. As such, it forms a foundation for those tasks you must perform after you have an operational network. Although many people think that system administration consists of simply designing and installing a network and then adding the users and groups, you are about to find out that much more must be considered during this process.

QUERYING ACTIVE DIRECTORY

Active Directory is nothing more than a database. When you realize this simple fact, you can consider all the tasks you usually perform with a database, and you can begin to use Active Directory to help you with your administrative tasks. One of the most common tasks performed against a database is searching it for data.

Active Directory stores data about each object on your network, including your users, groups, and computers. These objects have unique and distinct names, and they also have attributes. It is possible to query Active Directory for this data either with the tool provided by Microsoft or by using a third-party program or script. In this chapter, we will take a look at one of the built-in tools that ships with Windows 2000. This tool allows you to search for objects within Active Directory.

Before we examine the tool itself, however, we should first describe some of the most common objects stored in Active Directory. This list is not exhaustive, partly because Active Directory is **extensible**—you can create your own objects within Active Directory. We will concern ourselves here only with the default object types that are created. The general principles of querying for custom objects are the same as those outlined.

Common Objects

When we refer to **common objects**, we are talking about objects that have been defined in the Active Directory schema. The initial set of objects has been defined by Microsoft. If an object exists in Active Directory, then every domain controller in the enterprise will have an entry that represents it. Therefore, queries can be run against any copy of the Active Directory (or domain controller). This feature makes searching for data very efficient, because queries do not have to cross your network to be resolved.

When you add a resource to Active Directory, you create an object to represent it. Every resource that has been added to Active Directory is an object, and can therefore be returned as part of a search request. Table 7-1 lists the common objects that are created. Along with the name of the resource, the table gives a brief description of what the object represents.

Table 7-1 Common objects in Active Directory

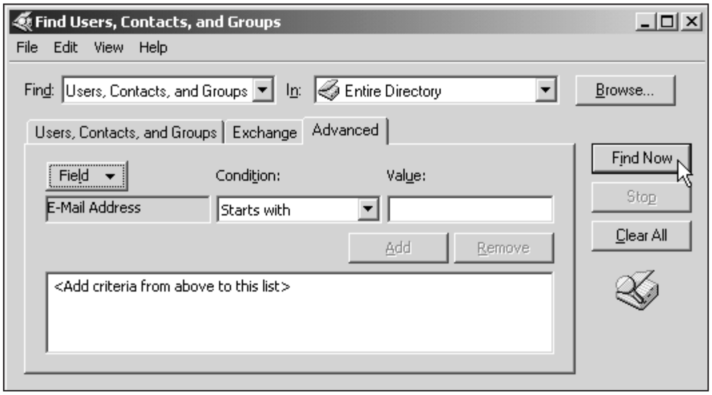
Object Type	Description
User Account	This object represents a user who has the ability to log on to a Windows 2000 network. You can search for the object (or user name) or you can use some of the optional fields available for this object, such as Home Address, Employee ID, Email Address, or Title.
Contact	This object type defines a person who has a connection with an organization, such as a supplier. Along with the contact name, you can also search for optional data such as Company Name, Division, or Fax Number.
Group	A group represents a grouping of user accounts, computers, or other groups. Groups are used for administrative purposes or security assignments. Along with group names, you can also search for optional properties such as Description, Members, Web Page Address, and Office Location.
Computer	This objects stores information about computers that are part of the domain. Along with the computer name, you can search for optional information such as the role the computer plays in the domain (Workstation, Server, or Domain Controller). You can also query specific fields such as Operating System, Operating System Version, and Managed By.
Printer	This object is a printer on your network. Printers that are added to Windows 2000 Professional systems must be manually added to Active Directory; printers added to a domain controller are automatically added. Along with a printer name, you can search for optional fields and features. Fields include Asset Number, Contact, Location, and Model. Additional features include the ability to print double sided, print in color, or staple.
Shared Folder	This object gives a pointer to a shared folder. It is only a pointer, because shared folders exist in the Registry of the computer where they are made. Therefore, a shared folder pointer in Active Directory points to the machine and entry only, not to the actual shared folder. Along with the name of a shared folder, you can also search for optional fields such as Description, Network Path, and Managed By.
Organizational Unit	An Organizational Unit (OU) is used to organize other Active Directory objects for administrative purposes. Along with the OU name, you can also search for optional fields such as Description and Managed By.

These common objects will probably make up most of your searches within Active Directory. Knowing the different types of objects is one part of the equation; without an easy-to-use interface to Active Directory, however, it will still be difficult to get the data you want. With this is mind, Microsoft has provided a Find dialog box in which you can quickly enter the data you want to search for and see the results returned.

Finding files and folders has been a common feature in Microsoft operating systems for quite some time. Microsoft has taken this familiar interface and extended it to accept the optional parameters and attributes of Active Directory objects. By using this interface, you can quickly search for objects within the directory.

Finding Active Directory Objects

You can quickly find objects within Active Directory by using the Find dialog box. This dialog box resembles the search tools used to search local hard disks in previous versions of Microsoft Windows 9.*x* and Microsoft Windows NT. This dialog box is shown in Figure 7-1.



Find: Users, Contacts, and Groups

In: Entire Directory

Field: E-Mail Address

Condition: Starts with

Value:

<Add criteria from above to this list>

Figure 7-1 The Find dialog box

The Find dialog box's appearance may change slightly depending on the type of object you are searching for. For instance, when you choose Users, Contacts, And Groups in the Find drop-down box, an additional Exchange information tab allows you to limit the search to Microsoft Exchange recipients. Other object types might add items or remove certain options.

Table 7-2 defines and describes the various options available within the Find dialog box. The list given is extensive and attempts to offer you an explanation for every possible option. This dialog box is deceptively simple.

Table 7-2 The Find dialog box's options

Option	Description
Find	The Find drop-down list allows you to choose the object type you wish to search for. These object types include users, computers, groups, printers, shared folders, and more, as defined in Table 7-1. One of the most important options is Custom Search, which allows you to search for a combination of object types using Lightweight Directory Access Protocol (LDAP).
In	This drop-down box allows you to limit the scope of your search. You can choose to search the entire directory, a specific domain, or an OU.
Browse	The Browse button displays a hierarchical list of the folders and objects available. You can then specify a specific path to search.

Table 7-2 The Find dialog box's options (continued)

Option	Description
Advanced tab	This tab is context sensitive, meaning that the contents of this tab will change depending on the object type you have elected to search. This tab displays many different options, which are defined in the following entries of this table. You can use the Advanced tab to construct queries based on drop-down boxes or to enter queries manually. In order to do so, you must be familiar with LDAP queries. (A discussion of LDAP queries is beyond the scope of this book.)
Field	Located on the Advanced tab, the Field button displays a list of optional fields that are available to be searched. The list of fields will vary depending on the object type you have elected to search. Some of these optional fields were listed earlier in Table 7-1.
Condition	The Condition drop-down list is located on the Advanced tab. Once you have selected a specific field to search, you can further limit the search by defining a condition. Condition elements include Starts With, Ends With, Is Not, and Not Present.
Value	The Value text box is located on the Advanced tab. If you use a specific attribute to limit your search, then you must also enter a specific value to search. You don't need to use wildcards in this field. For instance, if you want to search for all user objects that start with the letter *A*, you can simply choose Starts With in the Condition field and enter "A" in the Value field.
Search Criteria	The Search Criteria box is located on the Advanced tab. This text box contains each of the search options you have defined. A single search can contain a combination of custom searches. As you create these custom searches, they are listed in this box.
Find Now	This button causes the search to occur.
Stop	This button causes a search to stop. You might use the Stop button if the search you created is taking too long (probably meaning it is too broad in scope) or if the search results being displayed are not what you want. When you click on the Stop button, the results returned up to that point are displayed. You should note that this may not be a complete list of success matches for your criteria.
Clear All	Once you have completed a specific search with custom criteria, you may want to perform a second search. In order to do this, you should first clear the criteria you configured previously. You do so by clicking on the Clear All button.
Results	When you perform a search, the Results window is added to the Find dialog box. This window displays the results of your search. If your search's results are too large to fit in the window, scrollbars appear and allow you to scroll through the result set. Columns are returned along with the result set. By clicking on the column names, you can sort the result set.

The Find dialog box offers a relatively simple interface for searching Active Directory. By using a combination of standard searches and custom searches through the Advanced tab, you should be able to find just about any piece of information you want.

Permissions and Active Directory Objects

Administrators of Microsoft Windows NT 4 will be familiar with the concept of **permissions**. Permissions are used extensively in NTFS to grant or deny access to files and folders at the user or group level.

Windows 2000 and Active Directory extend this model to the object level. As a result, you can assign permissions to perform tasks at a specific object or attribute level. This process is known in Windows 2000 as **access control**. Each object within Active Directory has a security descriptor. This descriptor allows users or groups to be assigned specific permissions to an object. In the following sections, we will investigate this feature and how it can help you as a Windows 2000 administrator.

Introduction to Active Directory Permissions

You must consider two things when assigning permissions to objects: who should be allowed to access an object; and what permissions, or **actions**, users should have once they have gained access to an object.

You will find that assigning permissions to Active Directory objects is a balance between ease of use and the amount of administrative work for which you want to be responsible. For instance, it is possible to assign specific permissions to every object within Active Directory. However, although doing so would make the objects highly secure (you would have examined each object and assigned only the precise permission required), it would also increase the overhead of supporting your Active Directory implementation—almost certainly beyond what would be considered reasonable.

You must consider two levels: Active Directory security and object permissions. These levels work in tandem to define the security model that best fits your organization. Let's define these two terms in a little more detail.

Active Directory Security

You use Active Directory permissions to determine who does or does not have access to an object. By denying a user or group permissions on an object, you are effectively hiding this object from them. Only administrators or object owners can assign permissions on an object, thereby preventing unauthorized users from assigning permissions.

The list of users and groups that are allowed access to an object is stored in the object's Access Control List (ACL). Each object has an ACL. The ACL lists everyone who has been granted access to an object along with the actions that users or groups can perform.

One of the most common ways you will use ACLs is to assign administrative permission to OUs. In Windows 2000, it is possible to create an OU for a group of objects—perhaps users in the Finance department—and then to assign a user full administrative permissions over that OU. Doing so would have the effect of making the Finance department autonomous for day-to-day operations such as adding users or assigning permissions to shared folders within its own organization. Although some users might have administrative control over objects within that specific OU, they would not automatically gain administrative permissions over other objects. This arrangement keeps your Active Directory secure.

Object Permissions

Different types of objects have different permissions. Therefore, it is difficult to make blanket statements about what is and what is not allowed for a given object. We will take a look at some of the most common permissions in a moment.

You can assign object permissions at the specific user level or at the group level. If you decide to assign permissions at the user level, then you will increase the complexity of your security infrastructure. You should make sure that you document your permissions and that you keep this list in a secure place (perhaps on your hard disk using Encrypted File System [EFS]).

You may assign a specific user permissions on an object when the user also has default permissions from group membership. This situation can occur when you are using a combination of object permissions at the user and group levels. In such cases, the user will have a combination of both sets of permissions. For instance, if a specific user has the right to reset a password on an object, and at the same time he is a member of a group assigned permission to change a user name, the user will have a combination of both permissions—he has the ability to both reset the password and rename the object.

The one exception to the combination-of-permissions rule occurs when you deny access to an object. If a user is a member of a group, and you assign that group Full Control permissions on an object, then the user has Full Control through group membership. However, if you then specifically deny that user access to the object, the Deny permission overrides any other permissions. The net result is that the user does not have any permissions over the object. It is important to remember that Deny overrides all other permissions.

Be sparing in your use of Deny. It can be difficult to troubleshoot problems if you have been busy denying access to certain objects at the user level. It is almost always better to assign or deny permissions at the group level. You will have to create far fewer sets of permissions this way, and undoing mistakes and making changes to your security plan are much simpler if you use groups.

Here's one final caveat: It is possible for you to deny everyone access to an object. As a result, no one will have access to the object—including administrators. The object is effectively orphaned until an administrator takes ownership of it. Taking ownership of an object should be done sparingly; make sure you understand the consequences of performing an action before you change permissions on an object.

Permissions and Special Permissions

Active Directory uses two sets of permissions: standard permissions and special permissions. **Standard permissions** are the most common set of permissions assigned to a user or group. **Special permissions** offer a finer degree of control and are used infrequently. Using special permissions will increase your administrative overhead.

Each object type has its own set of permissions. For instance, a shared folder allows you to assign permissions for users to execute files and traverse the folder structure. This permission is not available for User objects because these actions are not applicable.

Rather than attempt to list every possible permission available, we will define a few of the most common permissions that can be assigned. Table 7-3 lists the permissions that are considered standard.

Table 7-3 Standard permissions

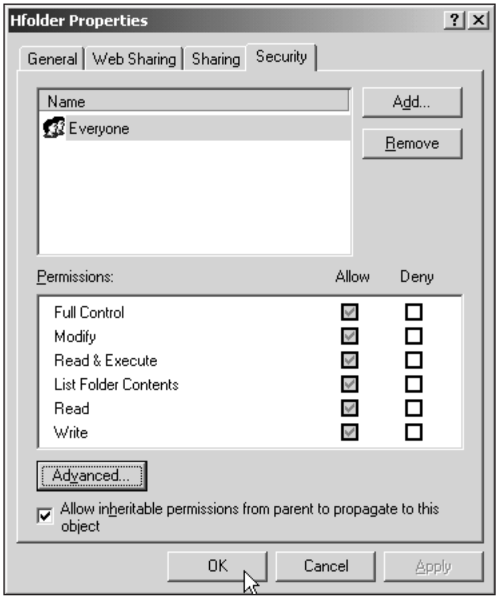
Permission	Description
Full Control	Assigns all the standard permissions along with Full Control and Take Ownership. It should be used sparingly.
Read	Enables users or group members to view the object and its attributes. This permission also lets you view the object owner and any permissions that have been assigned in Active Directory.
Write	Allows the user or group member to change the attributes of an object. Note that this permission does not assign Full Control or Take Ownership.
Create All Child Objects	When assigned as a permissions on a OU, allows the user or group member to create objects as children of that OU. Doing so creates a hierarchy of OUs or objects.
Delete All Child Objects	Allows the user or group member to delete objects from within a specific OU. This permission is most often assigned along with Create All Child Objects so that objects can be both created and removed.

Assigning Permissions

Now that you have an idea of some of the permissions that can be assigned and what they do, we can examine how a system administrator assigns permissions to a user or group. This is a common task that you will perform when working with Windows 2000.

You can assign permissions in several different places, but the end result is the same. For instance, to assign permissions to a User object or OU within Active Directory, you use the Active Directory Users and Computers console. On the other hand, to assign permissions to a shared folder, you use Explorer. Which tool to use largely depends upon the types of objects the tool lists. Explorer does not list users and groups—therefore you cannot assign permissions on those objects using that tool.

Figure 7-2 shows a typical Properties dialog box. In this case, we have chosen to show you the standard permissions that can be assigned to a folder. Note that the list of permissions given here is limited.



Name
Everyone

Permissions:	Allow	Deny
Full Control	☑	□
Modify	☑	□
Read & Execute	☑	□
List Folder Contents	☑	□
Read	☑	□
Write	☑	□

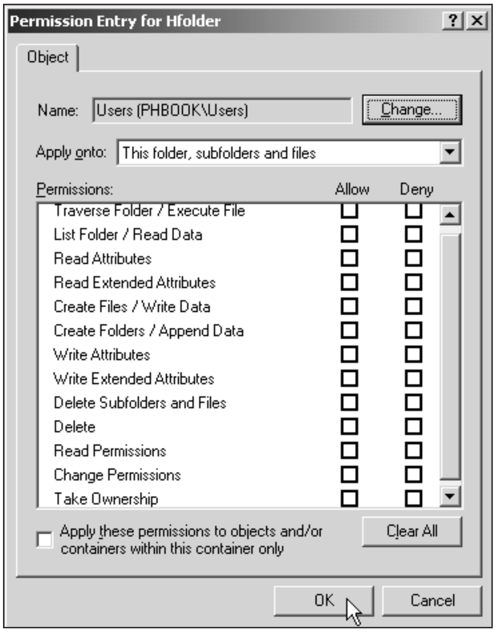
☑ Allow inheritable permissions from parent to propagate to this object

Figure 7-2 Standard permissions for a folder

The list of permissions shown in Figure 7-2 is a subset, also known as the **standard permissions** for this object type. By clicking on the Advanced button and choosing a group or user to assign permissions to, you can also view the advanced permissions for this object type (see Figure 7-3). Notice that the standard permissions are not duplicated in this list, and that many more options are available.

The checkboxes next to the permissions allow you to either grant or deny permissions to the selected user or group. Windows 2000 uses **inheritance** extensively. If the checkbox in the Allow or Deny column is grayed out, then that permission has been inherited from a parent object. (We will take a closer look at inheritance in a moment.) If you wish to prevent an object from inheriting permissions from its parent, you should uncheck the Allow Inheritable Permissions From Their Parent To Propagate To This Object checkbox (not shown in the figure).

To fully understand the net effect of assigning permissions, you should have a good understanding of inheritance. Let's take a look at inheritance and how it works.



Name: Users (PHBOOK\Users)

Apply onto: This folder, subfolders and files

Permissions:	Allow	Deny
Traverse Folder / Execute File	☐	☐
List Folder / Read Data	☐	☐
Read Attributes	☐	☐
Read Extended Attributes	☐	☐
Create Files / Write Data	☐	☐
Create Folders / Append Data	☐	☐
Write Attributes	☐	☐
Write Extended Attributes	☐	☐
Delete Subfolders and Files	☐	☐
Delete	☐	☐
Read Permissions	☐	☐
Change Permissions	☐	☐
Take Ownership	☐	☐

☐ Apply these permissions to objects and/or containers within this container only

Figure 7-3 Advanced permissions

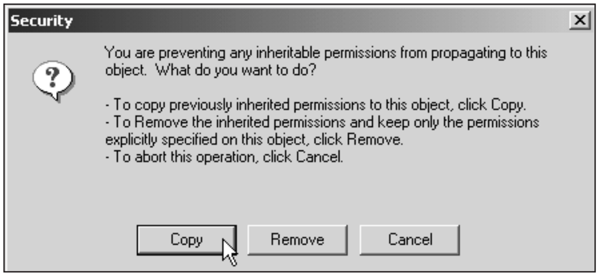
Permission Inheritance

Permissions are inherited by child objects by default. As a result, if you have a hierarchy of objects and you assign permissions to the object at the top of the hierarchy, all objects beneath it will be assigned the same permissions. This behavior can be overridden. It works much the same as inheritance works with files and folders. Inheritance was designed to simplify the administration of objects—by assigning a default set of permissions in a single place, all objects can be assigned the same default set.

Permission inheritance is useful with OUs. OUs are designed to simplify administration; you can achieve this goal by grouping together objects that have similar administrative needs, such as printers. By grouping all printers into a single OU called Printers, you can easily assign permissions to a group of users. For instance, if all printers are part of a single OU, you can create a group called Printer Administrators and assign that group Full Control permissions on the Printers OU. Through inheritance, all Printer objects within the OU will inherit this permission, and all members of the Printer Administrators group will have the Full Control permission.

We spoke earlier of the Allow Inheritable Permissions From Their Parent To Propagate To This Object option, which allows you to block inheritance from a parent object. This option lets you prevent the flow of permissions. It gives you a finer degree of control over objects, while at the same time increasing the amount of administrative overhead; it should therefore be used sparingly. When you uncheck this checkbox, you are presented with the Security dialog box shown in Figure 7-4.

As you can see in Figure 7-4, you are presented with three choices. The first choice is to copy the permissions from the parent object; this option means that the starting point for permissions for this object will be whatever is inherited from its parent. The second option is to remove the inherited permissions; this option allows you to remove the default permissions and to apply your own set. Finally, you can cancel the operation.



Security

You are preventing any inheritable permissions from propagating to this object. What do you want to do?

- To copy previously inherited permissions to this object, click Copy.
- To Remove the inherited permissions and keep only the permissions explicitly specified on this object, click Remove.
- To abort this operation, click Cancel.

Figure 7-4 The Security dialog box

The choice you make here will depend on the amount of work it will take to adjust the inherited permissions to match the set you wish the object to have. If you want to change only one or two settings, then it might be better to copy the settings. However, if the changes will be extensive, then you should remove the permissions and start from scratch. Understand that if you choose to remove permissions, the object will have no permissions set—by default, only administrators and the object's owner will be able to access the object.

Permission inheritance is an important concept, and you should take it into account when designing your security infrastructure. By using it wisely, with the maximum amount of inheritance left in place, you can design quite complex security plans with a minimum of work. That is not to say assigning individual permissions to an object is a bad thing to do—just be aware such a setup will take more planning and maintenance.

Making Resources Available

Adding objects to Active Directory and then assigning permissions to those objects are among the main functions of a system administrator. Active Directory stores information about objects and then returns the results of queries to users.

When Active Directory makes data available for display as query results, it is (in Windows 2000 terminology) *publishing* the data. **Publishing** means making the data available for viewing. You can publish many different kinds of data from Active Directory, including information about users, computers, printers, and files.

Some pieces of data are published automatically. For instance, User objects are published automatically so they can be returned in queries and used in groups. As you have seen, objects have a number of associated attributes.You should note that only a subset of attributes is published along with the object. The attributes that are automatically published include the most commonly searched-for items, including the login name of the user.

Other pieces of information either are not published at all or are published to a limited subset of users. For instance, security information about a User object is published but can be viewed only by administrators.

Publishing Resources

Some resources are published automatically, whereas others must be published manually. In order to publish manually, you must use the Active Directory Users and Computers console. Until a resource is published, users will not be able to see it on the network. Active Directory makes resources visible to your user community.

Don't forget, if a printer is installed on a domain controller, it is automatically published. If a printer is shared from a Windows 2000 Professional machine or a standalone server, then it must be published manually.You will be creating a share and publishing it in the Real-World Projects at the end of this chapter.

Publishing Network Services

Because Active Directory is so flexible, you probably will not be surprised to find that users, computers, and shared folders are not the only network resources that can be published. In fact, it's useful to make a host of other resources available for searching in Active Directory.

Imagine being able to search for a specific network service in Active Directory and then being able to administer that service once it is found. Doing so could save the administrator a lot of time. Of course, if you knew the server name, it might be just as easy to search for the server that is offering the service. By publishing the service itself, however, you can gain some administrative flexibility. In effect, you have divorced the service from the server; as a result, if the server that is offering the service changes, the service can still be found.

For example, you might publish the Certificate Services network service in Active Directory. This essential service requires periodic administration. By publishing the service, your administrators will be able to find it by querying Active Directory through the Active Directory Sites and Services Console. This process allows your administrators to concentrate on the service rather than on both the service and the server.

When to Publish Services

Before you publish every network service under the sun, let's examine some of the best reasons to publish this information. Once you understand the reasons that exist for doing so, you will be ready to design your own publishing criteria. You should publish the following kinds of data:

- *Data that is stable*—You should not integrate into Active Directory any data that will change constantly. The data you publish in Active Directory is replicated to every domain controller in Active Directory (see Chapter 14 for details). Because this replication can take some time to occur (with a maximum period of 15 minutes), it is possible for data returned from one replica of the Active Directory to be out of date. This situation will self-correct when replication takes place, but it is still an inconvenience. You should also take into account that replication requires network bandwidth. Data that changes frequently will generate additional network traffic, and the convenience of having the service published may be outweighed by the cost of network bandwidth. If you are not careful, services can appear and then disappear from Active Directory (if they are deleted). Also, users connect to some services via TCP/IP ports. If a port changes and this change has not had time to replicate to all domain controllers, then it might appear to users and administrators as though the service is not available.
- *Properties with a small amount of data*—Do not replicate large pieces of data within Active Directory. You must consider not only the frequency of a change, but also the size of the changed data. Active Directory will replicate data based on the entire property, not on part of a property. If a Description property is 50 bytes long and you change one word, then the whole 50 bytes must be replicated.
- *Useful information*—The information that you decide to replicate should have widespread significance. That is, replicate only the data that will be of some use to a large number of clients. If you have a property such as Asset Number, and that asset number is used by only five people in a network of thousands, then it makes little sense to replicate this data to every domain controller. Choose data that is commonly searched by a large number of people. Doing so ensures that you get the maximum benefit from the necessary network bandwidth and disk space.

Moving Objects

Objects within Active Directory can be moved to accommodate administrative tasks. For instance, you might have built a host of OUs around your departmental structure. User objects are stored within the respective departments' OUs. If a user is then transferred to a different business unit, you should move that User object from one OU to another.

Some significant differences exist between moving objects *within* a domain and *between* domains. We will take a look at each task in turn. When you're moving large numbers of objects, consider using a scripting engine such as Windows Scripting Host (WSH). WSH is discussed in more detail in Chapter 11.

Moving Objects within a Domain

Moving objects within a domain is a fairly simple task. Most commonly, you might do this when you want to move a user object from one OU to another OU—perhaps during an organizational restructuring. You use the Active Directory Users and Computers console to perform this task.

When moving objects within a domain, take the following factors into account:

- Permissions will be inherited from the new OU. The permissions that were inherited from the previous OU will no longer be applied. This change can affect access to shared folders and administrative permissions to other user objects.
- Permissions that have been assigned to the specific object will be retained. This fact can be both a blessing and a curse. It also illustrates perfectly our earlier point regarding the complexity of administration when you assign permissions at the object level. If you no longer want the User object to have a specific permission, then you must remove the permission from the object after the object has been moved. It is better to assign permissions at the OU level, so a user is automatically assigned permissions.
- It is possible to move multiple objects at the same time. To do so, simply highlight one of the objects you want to move, hold down the Ctrl key, and select additional objects.
- You must initiate the move on the domain controller acting as the relative ID (RID) master of the domain that currently contains the object.

Moving Objects between Domains

Moving objects between domains is a little more complicated, partly because Microsoft has failed to provide us with a Graphical User Interface (GUI) for this purpose. More significantly, though, moving an object from one security boundary to another is technically complex.

In order to move an object from one domain to another, you must use a command-line utility called MoveTree. This utility is not installed by default. In fact, it is sometimes preferable to delete an object and re-create it rather than move it. In order to install the MoveTree utility, you should execute the Setup program in the \Support\Tools folder on your Windows 2000 CD.

Moving an object within a domain is a fairly simple process; however you must consider many more factors when moving objects from one domain to another. An example is a security identifier (SID) that is assigned to an object when it is created. An object's SID is based partially upon the domain in which it exists. If the domain then changes, the SID is incorrect. Windows 2000 will go ahead and create a new SID for the object and store a copy of the old SID information in a new field called SIDhistory. Whenever a

User object logs on to a Windows 2000 network, the current SID and all entries in the SIDhistory field are included in the security token for the object. As a result, the object retains some of its permissions even though it has been moved.

Each object in a Windows 2000 network also has a unique Globally Unique Identifier (GUID), which is a reference number to an object. Because this number is universally unique, you don't need to change it—moving an object from one domain to another will not affect the object's GUID.

It is also possible to move other types of objects, such as OUs. When you're moving OUs, any Group Policy Objects (GPOs) that have been assigned to the OU will remain intact. This feature can be useful when you want the same settings to be applied. The GPO data is not replicated to the new domain controllers, however. As a result, the data for the GPO remains in the old domain and must be pulled from there. You should consider the traffic associated with this data transfer and with re-creating Group Policy settings in the new domain.

As you can see, MoveTree supports many different types of operations. Some additional operations you might want to consider are as follows:

- You can move an object or objects between domains even if the object contains child objects. This action is supported only if you are moving objects within the same Active Directory forest.
- You can move both local groups and global groups between domains if they do not have any members. If a group contains members, however, it can be moved only between containers within the *same* domain.
- You can move Universal groups both within and between domains, regardless of whether the group has members.

MoveTree Restrictions

Some restrictions come with the list of tasks that can be performed with the MoveTree utility. Some operations are considered unsupported, which is another way of saying they cannot be performed successfully.

Let's take a look at some of the objects that cannot be moved using the MoveTree utility:

- Local and global groups that have members.
- Some object data. This data includes users' personal data (documents and spreadsheets), their encrypted files, and public key certificates. Logon scripts are not copied, either. As a result, a user being moved from one domain to another may end up having a very different user experience. Keep this fact in mind when you plan to move users.
- System objects. Any objects that belong to the system cannot be moved, including any objects stored within the Configuration or Schema containers.

- Objects that are stored in special containers such as LostAndFound, Builtin, and System.
- Domain controllers. If you want to move a domain controller, you must demote the domain controller and then promote it into the new domain using the DCPromo utility.
- Any object with the same fully qualified domain name. Every object within Active Directory must have a unique name within the hierarchy. If an object has the same name, then the move will fail.

Along with these restrictions, you should be aware of some significant items regarding the moving of User and Group objects:

- When you're moving a User or Group object, the object must not have any objects beneath it. Such an object is known as a **leaf object**.
- If the User or Group object's name already exists in the context to which you are trying to move it, then the move operation will fail. You should also note that some security settings can also cause the move to fail. An example would be the minimum password length setting. If the User object does not have a password of at least the minimum length, the move operation fails.
- If the User object is a member of any global group, then the move operation fails because a global group can contain members only from its own domain. The one exception is the Domain Users global group when it is the primary group for the User object (default behavior). User objects are automatically added to the group when they are created, and can therefore be removed.

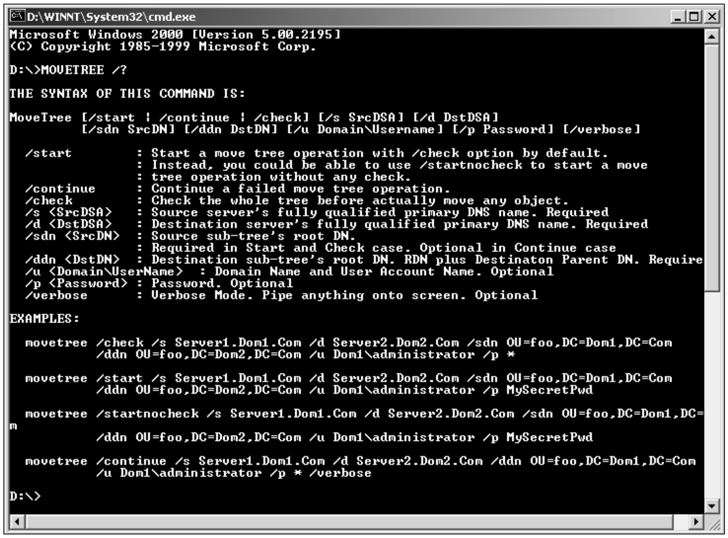
Using the MoveTree Command

The MoveTree utility is run from the command line. Currently, no GUI is available for this utility. That does not mean the utility is not feature rich—in fact, this command-line utility is a boon. You can include it quite easily in batch files or scripts.

Figure 7-5 shows the **MoveTree** command when run with the **/?** parameter. Notice that this command displays all the available options with descriptions and examples.

MoveTree generates three log files when it is run; these files are stored in the folder where the command is run. The log files supply you with a list of errors and statistics and can be useful for troubleshooting possible problems. The three files are:

- *MoveTree.err*—Lists errors that were encountered.
- *MoveTree.log*—Lists statistics for the move operation.
- *MoveTree.chk*—Lists any errors that have been found during the test phase of the command. This log can alert you to possible errors that might occur, so you can address them before performing the actual command. Use the **/check** option to create this file.



```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

D:\>MOVETREE /?

THE SYNTAX OF THIS COMMAND IS:

MoveTree [/start | /continue | /check] [/s SrcDSA] [/d DstDSA]
         [/sdn SrcDN] [/ddn DstDN] [/u Domain\Username] [/p Password] [/verbose]

  /start          : Start a move tree operation with /check option by default.
                  : Instead, you could be able to use /startnocheck to start a move
                  : tree operation without any check.
  /continue       : Continue a failed move tree operation.
  /check          : Check the whole tree before actually move any object.
  /s <SrcDSA>     : Source server's fully qualified primary DNS name. Required
  /d <DstDSA>     : Destination server's fully qualified primary DNS name. Required
  /sdn <SrcDN>    : Source sub-tree's root DN.
                  : Required in Start and Check case. Optional in Continue case
  /ddn <DstDN>    : Destination sub-tree's root DN. RDN plus Destinaton Parent DN. Require
  /u <Domain\UserName>  : Domain Name and User Account Name. Optional
  /p <Password> : Password. Optional
  /verbose        : Verbose Mode. Pipe anything onto screen. Optional

EXAMPLES:

  movetree /check /s Server1.Dom1.Com /d Server2.Dom2.Com /sdn OU=foo,DC=Dom1,DC=Com
           /ddn OU=foo,DC=Dom2,DC=Com /u Dom1\administrator /p *

  movetree /start /s Server1.Dom1.Com /d Server2.Dom2.Com /sdn OU=foo,DC=Dom1,DC=Com
           /ddn OU=foo,DC=Dom2,DC=Com /u Dom1\administrator /p MySecretPwd

  movetree /startnocheck /s Server1.Dom1.Com /d Server2.Dom2.Com /sdn OU=foo,DC=Dom1,DC=
m
           /ddn OU=foo,DC=Dom2,DC=Com /u Dom1\administrator /p MySecretPwd

  movetree /continue /s Server1.Dom1.Com /d Server2.Dom2.Com /ddn OU=foo,DC=Dom1,DC=Com
           /u Dom1\administrator /p * /verbose

D:\>
```

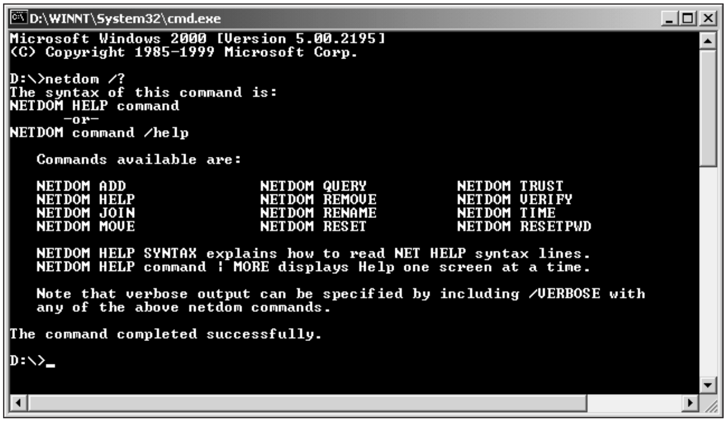
Figure 7-5 The MoveTree command-line utility

The MoveTree utility is useful for moving users, groups, and OUs between domains. You might also want to move other significant objects, however. These objects cannot be moved with MoveTree—instead, you must use a different utility. This utility is discussed in the next section.

Moving Workstations and Member Servers

When you install support tools such as MoveTree, as outlined in the previous section, you also install a command-line utility called Netdom. This utility is used to move workstations and member servers between domains. Use of this command-line utility is very much the same as we saw with the MoveTree utility.

The command-line options and syntax are shown in Figure 7-6. You can get more detailed information from the Windows 2000 Resource Kit help file. Unfortunately, detailed information about the Resource Kit utilities is outside the scope of this book. However, it is highly recommended that any administrator of Windows 2000 purchase the full kit; it contains a host of utilities and information.

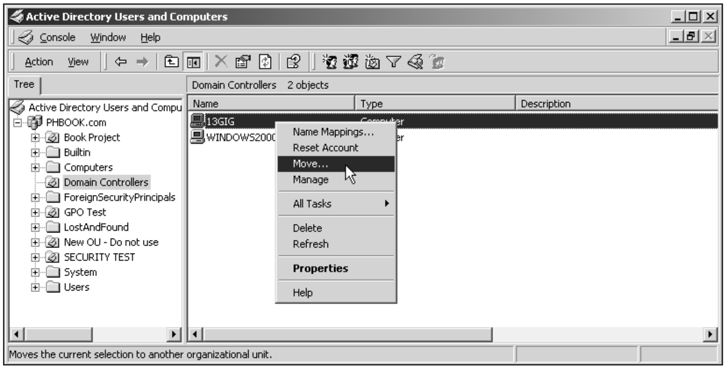


```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

D:\>netdom /?
The syntax of this command is:
NETDOM HELP command
      -or-
NETDOM command /help

   Commands available are:

   NETDOM ADD              NETDOM QUERY           NETDOM TRUST
   NETDOM HELP             NETDOM REMOVE          NETDOM VERIFY
   NETDOM JOIN             NETDOM RENAME          NETDOM TIME
   NETDOM MOVE             NETDOM RESET           NETDOM RESETPWD

   NETDOM HELP SYNTAX explains how to read NET HELP syntax lines.
   NETDOM HELP command | MORE displays Help one screen at a time.

   Note that verbose output can be specified by including /VERBOSE with
   any of the above netdom commands.

The command completed successfully.

D:\>_
```

Figure 7-6 The Netdom command-line utility

Moving Domain Controllers between Sites

The easiest move operation involves moving domain controllers between sites. In the Real-World Projects at the end of this chapter, you will move a shared folder from one location in Active Directory to another. You can use this same technique to move domain controllers between sites. In short, you can simply right-click on a domain controller object within the Active Directory Users and Computers console to bring up the context-sensitive menu; then select the Move command. This command (shown in Figure 7-7) enables you to choose the container to which the domain controller will be moved.

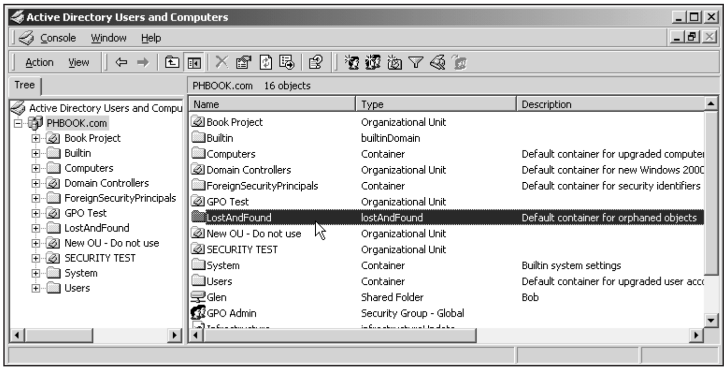


Moves the current selection to another organizational unit.

Figure 7-7 The context-sensitive menu with the Move command

Moving Objects to the LostAndFound Container

Some objects cannot be successfully moved from one domain to another. For example, some child objects will fail to move. If a parent object of a child is moved, then the parent is no longer available, and the child is considered to be **orphaned**. An orphaned object is moved to the LostAndFound container, which is visible in the Active Directory Users and Computers console. You should periodically take a look at the LostAndFound container and move any objects you want to keep into another parent object. The LostAndFound container can be seen in Figure 7-8.



Active Directory Users and Compu
PHBOOK.com
Book Project
Builtin
Computers
Domain Controllers
ForeignSecurityPrincipals
GPO Test
LostAndFound
New OU - Do not use
SECURITY TEST
System
Users

PHBOOK.com 16 objects

Name	Type	Description
Book Project	Organizational Unit	
Builtin	builtinDomain	
Computers	Container	Default container for upgraded compute
Domain Controllers	Organizational Unit	Default container for new Windows 200
ForeignSecurityPrincipals	Container	Default container for security identifiers
GPO Test	Organizational Unit	
LostAndFound	lostAndFound	Default container for orphaned objects
New OU - Do not use	Organizational Unit	
SECURITY TEST	Organizational Unit	
System	Container	Builtin system settings
Users	Container	Default container for upgraded user acc
Glen	Shared Folder	Bob
GPO Admin	Security Group - Global	

Figure 7-8 The LostAndFound container

By default, you will not be able to see the LostAndFound container in the Active Directory Users and Computers console. In order to see this container, you must be in Advanced Features view. To enter this view, choose View | Advanced Features in Active Directory Users and Computers.

DELEGATING AUTHORITY

Wouldn't it be nice if some of the more routine tasks that a system administrator performs could be offloaded to someone else? Although it can sometimes be tempting to make sure that no one but the administrator can perform certain tasks, doing so means that everything must go through that one person or team—and often the team becomes a bottleneck.

What's more, some administrative tasks can be quite routine. If you could assign permissions to perform these tasks and be assured that no other permissions were being assigned, wouldn't it be nice to let a departmental staff member add Printer objects to and remove them from Active Directory?

This is the goal of **delegation of control**: Certain administrative tasks are assigned to a user or group. The interface for performing the delegation task is the Delegation Of Control Wizard. This wizard has several options, which we will look at in a moment.

The following list shows the types of tasks that can be assigned with the Delegation Of Control Wizard:

- Allow a User object the permissions to change the properties of a particular container, including such items as the container name or description.
- Allow a User object to create, delete, or modify objects in a particular OU or container. You can set this feature at the object-type level, which means you can allow a User object to create printers but not user accounts, or vice versa. Note that the ability to assign this permission at the OU level means you can give departments within your organization a level of administrative autonomy.
- Allow a User object to modify the properties of objects in an OU or container.

Notice that the implications of these tasks are quite far reaching. In large organizations, creating and deleting User objects for temporary workers can generate a lot of administrative work. By delegating authority for these types of functions, the administrators can concentrate their efforts on large, more complex efforts.

As we saw earlier, you should be careful when delegating authority. What starts out as a method of reducing the amount of work for your IT administrative staff may end up creating more work for them. For instance, delegating authority at the object level (rather than at the container or OU level) can cause an increase in the amount of administrative overhead, because someone will have to track what has been delegated and to whom.

It is far better to delegate authority at the container or OU level; however, doing so can have effects that are farther reaching. If you delegate the authority to create objects of a certain type in an OU, then the User object can create as many objects as it wants. Make sure that the user who will get this new authority has been well trained in the consequences of performing an action.

You should take delegation of authority into consideration when designing your OU structure. Your OU design should match your administrative needs, and delegation of authority is likely to be a large part of your future plans. Delegation allows you to decentralize some of the administrative work that takes place in every organization. Although you must strike a balance when delegating (you do not want to allow departmental administrators to have permissions outside the realm of their responsibilities), delegation can have many benefits when used wisely.

If you are going to upgrade to Windows 2000, you may be consolidating domains. In Microsoft Windows NT 4, you may have had more than one domain. In Windows 2000, it is possible—and even desirable—to have a single domain. In the process, you must face a political issue: You should severely restrict the members of the Domain Admins and Enterprise Admins groups. Staff members who were previously members of Domain

Admins may feel that their jobs have been reduced in importance. You can alleviate this situation by using delegation of authority to give them Full Control permission over an OU that contains all objects relating to their line of business.

Follow these guidelines when using delegation of authority:

- Always assign permissions at the OU or container level. It is possible to assign permissions at the object level, but doing so is discouraged.
- If you want to assign permissions at the object level, then you cannot use the Delegation Of Control Wizard. Instead, you must use Active Directory Users and Computers console.
- Maintain a record of who has been assigned permissions. Windows 2000 does not do a good job of recording what has been assigned. Although you can always get the properties of an OU or container, doing so can be cumbersome when you're searching multiple objects. Keep a record in a database and make sure it is available to all administrators.

Using the Delegation Of Control Wizard

As you will see, the Delegation Of Control Wizard is fairly straightforward. You should use the wizard whenever possible, because Microsoft has streamlined the commands required to make delegation work. In order to start the wizard, you must use the Active Directory Users and Computers console. For instance, if you want to delegate authority to an OU, you start the Active Directory Users and Computers Console and then right-click on the OU name in the right-hand panel. Doing so brings up the context-sensitive menu. Select Delegate Control to start the wizard, which is shown in Figure 7-9.

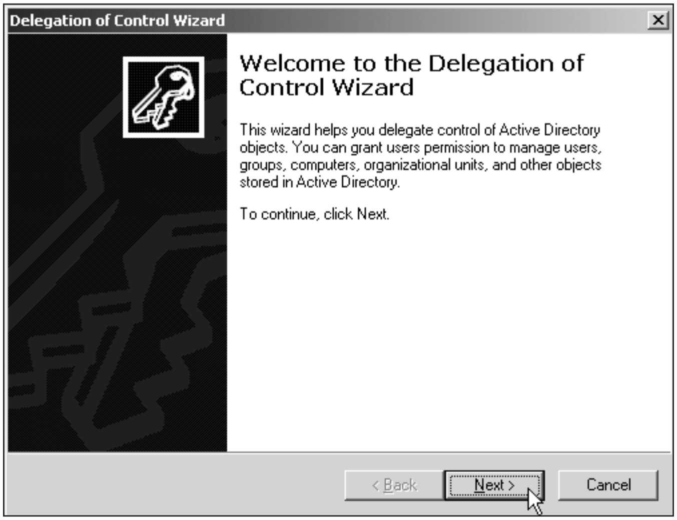
When you click on the Next button, you can select the users or groups to which you will delegate control. Be careful when selecting users and groups—make sure you limit the number of users that will be granted permissions for the OU or container.

Once you have selected the users or groups, click on Next again. This time, the wizard prompts you to enter the tasks to be assigned to the user or group, as shown in Figure 7-10.

Figure 7-10 shows the options when you're delegating common tasks such as creating and deleting groups or resetting passwords. If you want to assign more granular permissions, then click on the Create A Custom Task To Delegate button. (The options presented, if you do this, are outside the scope of this book.) Once you have selected your permissions, click on Next; the wizard will display a summary screen. That is all there is to the Delegation Of Control Wizard!

Note

Don't forget that all data affecting Active Directory must be replicated to every domain controller in the domain. As a result, a time lapse often occurs between performing a task—such as running the Delegation Of Control Wizard—and the permissions' actually being assigned. You should take this time lapse into account. It is not likely that permissions granted with this wizard will be immediately available. The permissions can take up to 15 minutes to be granted.

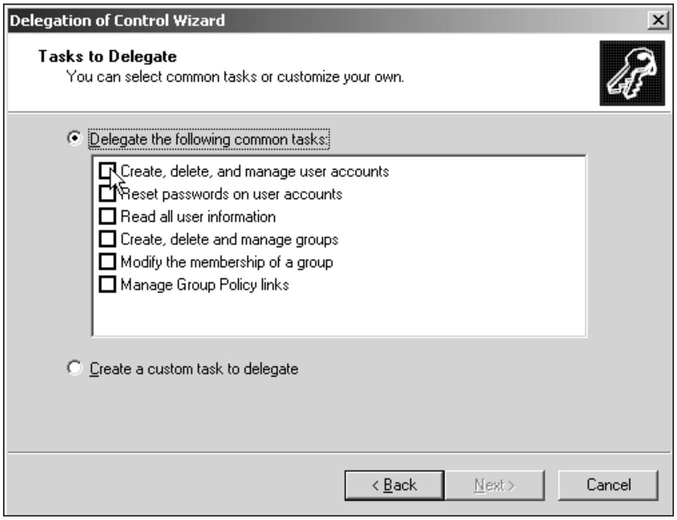


Welcome to the Delegation of Control Wizard

This wizard helps you delegate control of Active Directory objects. You can grant users permission to manage users, groups, computers, organizational units, and other objects stored in Active Directory.

To continue, click Next.

Figure 7-9 The Delegation Of Control Wizard



Tasks to Delegate

You can select common tasks or customize your own.

● Delegate the following common tasks:

- □ Create, delete, and manage user accounts
- □ Reset passwords on user accounts
- □ Read all user information
- □ Create, delete and manage groups
- □ Modify the membership of a group
- □ Manage Group Policy links

○ Create a custom task to delegate

Figure 7-10 The Tasks to Delegate screen of the Delegation Of Control Wizard

Delegating authority is a key feature of Windows 2000. When it's used in conjunction with a good OU design, you can save yourself a lot of work. However, don't forget that ultimately, the system administrators are responsible for the network. Do not overuse the Delegation Of Control Wizard, but don't forget about it, either.

Chapter Summary

- In this chapter, we looked at some of the aspects you need to be aware of when administering Active Directory. We started by examining ways you can query the data stored in Active Directory. You can query Active Directory for any data that it contains, if you have sufficient permissions to do so (some objects, such as system objects, are visible only to administrators).
- You learned that several common objects exist. These objects include User Accounts, Groups, Computers, and Printers. We also noted that you can create custom objects.
- We then discussed the Find dialog box. This dialog box can perform complex searches against Active Directory data. The attributes that are available to be searched vary among the different object types. You learned that although the most common options (such as object name) are displayed by default, you can also use the Advanced tab to access optional fields and narrow a search.
- We then discussed Active Directory permissions. Each object has a security descriptor that contains, among other things, a list of users or groups that have some level of control over the object. You learned that there are two types of permissions: Active Directory permissions and object permissions. The latter determine what action can be performed on an object, whereas the former determine whether an object is visible to a user or group.
- We saw that it is possible to allow or deny permissions on an object. When a user or group is denied access, this setting overrides all other assigned permissions. Other than the Deny permission, all permissions assigned to a user through direct assignment or group membership are cumulative.
- We took a brief look at some of the available permissions. There are two sets of permissions: standard permissions and special permissions. Standard permissions are commonly granted. Special permissions offer more granularity but also increase the complexity of your environment.
- We then examined permission inheritance. We saw that permissions for Active Directory objects operate in much the same way as permissions assigned to files and folders. When permissions are granted to a container object, the user or group gains rights over any Active Directory objects within that container. Also, any child container will, by default, inherit any permissions assigned to its parent container. It is possible to stop this from happening by blocking inheritance.
- Before network resources can be seen by users, they must be published. Some objects, such as printers installed on domain controllers, are published automatically. Others must be published manually. You also learned that you can

publish network services, such as Certificate Services. When doing so, you should exercise caution and publish only data that is fairly static and small. This is the case because Active Directory must replicate all data to domain controllers, and doing so takes time, processing, and bandwidth.

- We took a look at objects and how they can be moved within Active Directory. This task can be quite difficult when objects such as OUs and groups must be moved from one domain to another. We noted that the tools built into Windows 2000 work only when moving objects within the same tree or forest.
- You learned that moving an object within a domain is a fairly simple process using the Active Directory Users and Computers console. However, moving objects between domains requires you to install the support utilities from the Windows 2000 compact disk. (These utilities are not installed by default.) To move User, Group, and OU objects between domains, you must use the MoveTree command-line utility. To move workstations or member servers, you must use the Netdom command-line utility.
- Finally, we examined delegation of control, which allows you to delegate certain administrative tasks within a container object to users or groups. Doing so has the benefit of decentralizing day-to-day administrative tasks and reducing the workload of busy system administrators.
- We saw that you can delegate control quite simply by using the Delegation Of Control Wizard. This wizard walks you through the process of assigning common tasks to users and groups with a minimum of fuss. We explained that it is a good idea to be sparing with this feature, and to make sure that users who are given administrative permissions receive adequate training.